UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Instituto de Geociências e Ciências Exatas - IGCE
Curso de Bacharelado em Ciências da Computação

MAICON DALL'AGNOL

TRABALHO DE REGRAS DE ASSOCIAÇÃO

Professora: Dra. Adriane Beatriz de Souza Serapião

Rio Claro - SP

2019

1 Introdução

Este trabalho consiste em aplicar o conhecimento de regras de associação adquirido na disciplina Tópicos: Aprendizado de Máquina, tendo assim como objetivo:

- Escolha dois datasets específicos para a tarefa de Regras de Associação.
- Para cada dataset, você irá aplicar os algoritmos APriori e FP-Growth. Seu objetivo é minerar regras que contenham informações relevantes no dataset, seja porque alguma combinação de itens aparece com muita frequência, seja porque alguma combinação de itens não aparece com muita frequência mas está se destacando. Para isso, você deve variar os parâmetros de suporte, confiança e lift.
- Existem duas métricas não estudadas em sala de aula, 'leverage' e 'conviction', que estão disponibilizadas no pacote mlxtend para regras de associação. Estude essas métricas e explique como elas podem contribuir para as análises dos seus datasets.
- Compare os resultados gerados pelos dois algoritmos. Conclua sobre as diferenças encontradas nos resultados de cada dataset na aplicação dos dois algoritmos.

2 Desenvolvimento

Para o desenvolvimento das atividades inicialmente foram escolhidos duas bases bases de dados. A primeira base a ser utilizada corresponde a dados de *reviews* de um E-Commerce de Roupas Femininas contendo informações como idade, avaliação, categoria que o produto pertence, entre outras; A segunda base é composta dados educacionais que é coletado do sistema de gerenciamento de aprendizado, nela há dados como genero, nacionalidade, número de vezes que o aluno levanta a mão, entre outros dados.

2.1 Pré-processamento e Visualização

Ambas bases haviam dados categóricos e numéricos, na primeira base alguns atributos numéricos foram removidos e outros foram discretizados, ainda na primeira base também foram removidas transações que continham item (atributos=valor) faltantes; Para a segunda base visualizou através de um *boxplot* que os dados numéricos estavam variando de 0 a 100 com uma média diferente para cada atributo, desta forma todos os dados numéricos foram discretizados em categorias binárias (abaixo ou acima média do atributo).

Para a aplicação dos algoritmos também transformou-se os dados do formato dataframe para um array da biblioteca Numpy.

2.2 Regras de Associação

Para aplicação do algoritmo Apriori inicialmente transformou-se os dados para um formato transacional utilizando TransactionEncoder, dessa forma utilizou-se o algoritmo apriori implementado na biblioteca mlxtend para extração dos *itemsets* frequentes e $association_rules$ para extração das regras com confiança acima do corte.

Para aplicação Fp-Growth primeiramente implementou-se duas funções cujo propósito era: a. verificação de suporte e b. aplicação, cálculos das medidas. Nesta etapa algumas dificuldades foram encontradas uma vez que a implementação do algoritmo fp_growth baixada pelo indexador de pacotes de python PyPI não retornava todas as medidas, portanta algumas tiveram que ser calculadas o que gerou alguns problemas ao se buscar os suportes do antecedente e consequente.

2.3 Avaliação

Devido aos problemas encontrados na etapa de extração das regras a avaliação dos resultados também foi comprometida, contudo ainda sim avaliou-se os resultados obtidos .

Para a primeira base o em ambos algoritmos utilizou-se suporte mínimo de 20% e confiança de 80% com o intuito de pegar mais regras mas que tivessem uma melhor confiança. Das regras geradas algumas se destacam como quando um adulto avalia como muito bom, com uma altíssima confiança ele vai recomendar a loja. Outra regra que se destaca pela conviction infinita é Knits → Tops, contudo esta poderia ser uma regra redundante dentro do escopo da base uma vez que o antecedente trata-se do nome de uma classe de produto enquanto o consequente de um departamento, de forma que é possível ter uma classe de produto pertencente apenas a um departamento, não agregando conhecimento algum apesar das altas estatísticas.

Para a segunda em ambos algoritmos utilizou-se suporte mínimo de 35% e confiança de 80%. Nesta base algumas regras redundantes também foram geradas como o local do nascimento levando para a nacionalidade e vice-versa. Outras regras já são mais interessantes como (Under-7, $overmeanraisedhands$) → $overmeanVisITedResources$ com uma alta confiança, isso leva ao ponto que as crianças que faltam menos e levantam a mão mais vezes que a média também são as crianças que visitam os recursos mais vezes que a média. Outra regra interessante seria Yes → Good, mostrando que pais que responderam a pesquisa para a escola tem um nível de satisfação bom com ela. Nestas duas regras exemplificadas os valores de *leverage* ficaram em torno de 0.13, uma vez que está mais próximo de 0 do que de 1, pode-se concluir que essas regras estão indicando uma possível independência entre elas, contrastando com os valores de confiança obtidos.

Em teoria ambos algoritmos deveriam apresentar as mesmas regras para os mesmos valores de suporte e confiança mínima, contudo, devido algum possível problema na implementação, o algoritmo Apriori obteve mais regras que o FP-Growth na primeira base e menos na segunda base. Entretanto, para as regras que apareceram em ambos algoritmos todas as medições são iguais.

Para ambas bases os parametros foram alterados manualmente buscando um equilibrio entre um número não tão alto de regras e um bom valor de confiança a fim que pudessem ser facilmente visualizável. Para uma análise mais aprofundada, por exemplo, para segunda base, devido a presença de uma classe, poderia-se diminuir os valores de suporte e confiança buscando encontrar regras que tivessem como consequente os valores de classe de modo a gerar regras que pudessem ser usadas para classificar novas transações.

Regras

June 5, 2019

1 0. Introdução

Trabalho:
Aluno: Maicon Dall'Agnol
R.A.: 151161868
Disciplina: Tópico em Aprendizado de Máquina
Objetivos :

- Escolha dois datasets específicos para a tarefa de Regras de Associação. Não tente "produzir" um dataset como esse. Possivelmente, você irá falhar nisso. O dataset para Regras de Associação deve conter transações.
- Para cada dataset, você irá aplicar os algoritmos APriori e FP-Growth. Seu objetivo é minerar regras que contenham informações relevantes no dataset, seja porque alguma combinação de itens aparece com muita frequência, seja porque alguma combinação de itens não aparece com muita frequência mas está se destacando. Para isso, você deve variar os parâmetros de suporte, confiança e lift.
- Existem duas métricas não estudadas em sala de aula, 'leverage' e 'conviction', que estão disponibilizadas no pacote mlxtend para regras de associação. Estude essas métricas e explique como elas podem contribuir para as análises dos seus datasets. (Veja: http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/)
- No relatório, você deverá explicar como usou os parâmetros para a mineração e o que foi obtido. As regras selecionadas devem ser exibidas com suas medidas equivalentes (suporte, confiança, lift, leverage, conviction). Mostre as regras que você considerou mais relevantes e justifique por quê.
- Compare os resultados gerados pelos dois algoritmos. Conclua sobre as diferenças encontradas nos resultados de cada dataset na aplicação dos dois algoritmos.

1.1 0.1 Dependências

Para realização da tarefa foram utilizados as seguintes bibliotecas:

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
```

```
import seaborn as sns
import pandas_profiling

# Encoder
from mlxtend.preprocessing import TransactionEncoder

# Algoritmos
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
import pyfpgrowth

#Metricas
from mlxtend.frequent_patterns import association_rules

import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

2 1. Dados

Este é um conjunto de dados de E-Commerce de Roupas Femininas que gira em torno das avaliações escritas pelos clientes

2.1 1.1 Informações sobre os dados:

Atributos:

- Clothing ID: Integer Variável categórica que se refere à peça específica que está sendo revisada.
- Age: Positiva Variável inteira da idade dos revisores.
- Title: variável de string para o título da revisão.
- Review Text: variável de string para o corpo de revisão.
- Rating: Variável Integral Positiva Positiva para a pontuação do produto concedida pelo cliente de 1 pior, a 5 melhores.
- Recommended IND: Variável binária indicando onde o cliente recomenda o produto, em que 1 é recomendado, 0 não é recomendado.
- Positive Feedback Count: Integral Positivo documentando o número de outros clientes que consideraram este comentário positivo.
- Division Name: Nome categórico da divisão de alto nível do produto.
- Department Name: nome categórico do nome do departamento do produto.
- Class Name: nome categórico do nome da classe do produto.

2.2 Importando Dataset

```
In [2]: roupas = pd.read_csv('Womens Clothing E-Commerce Reviews.csv')

In [3]: roupas.head()
```

```
Out[3]:    Unnamed: 0  Clothing ID  Age                    Title  \
        0           0          767   33                      NaN
        1           1         1080   34                      NaN
        2           2         1077   60  Some major design flaws
        3           3         1049   50         My favorite buy!
        4           4          847   47         Flattering shirt

                                           Review Text  Rating  Recommended IND  \
        0  Absolutely wonderful - silky and sexy and comf...       4                1
        1  Love this dress!  it's sooo pretty.  i happene...       5                1
        2  I had such high hopes for this dress and reall...       3                0
        3  I love, love, love this jumpsuit. it's fun, fl...       5                1
        4  This shirt is very flattering to all due to th...       5                1

           Positive Feedback Count   Division Name Department Name Class Name
        0                        0       Initmates        Intimate  Intimates
        1                        4         General         Dresses    Dresses
        2                        0         General         Dresses    Dresses
        3                        0  General Petite         Bottoms      Pants
        4                        6         General            Tops    Blouses

In [4]: roupas.drop(columns=['Unnamed: 0', 'Clothing ID', 'Review Text', 'Review Text', 'Title

In [5]: roupas.isna().sum()

Out[5]: Age                 0
        Rating              0
        Recommended IND     0
        Division Name      14
        Department Name    14
        Class Name         14
        dtype: int64

In [6]: roupas.dropna(inplace=True)
```

2.3 Discretizando

```
In [7]: roupas['Age'] = ['adult' if x < 60 else 'elderly' for x in roupas['Age']]

In [8]: roupas['Recommended IND'] = ['yes' if x == 1 else 'no' for x in roupas['Recommended IN

In [9]: list_aux = []
        for element in roupas['Rating']:
            if element == 5:
                list_aux.append('very good')
            elif element == 4:
                list_aux.append('good')
            elif element == 3:
                list_aux.append('normal')
```

```
            elif element == 2:
                list_aux.append('bad')
            else:
                list_aux.append('very bad')

        roupas['Rating'] = list_aux
```

```
In [10]: roupas_np = roupas.to_numpy()
```

```
In [17]: for coluna in roupas.columns:
             print(roupas[coluna].unique(), '\n')
```

```
['adult' 'elderly']

['good' 'very good' 'normal' 'bad' 'very bad']

['yes' 'no']

['Initmates' 'General' 'General Petite']

['Intimate' 'Dresses' 'Bottoms' 'Tops' 'Jackets' 'Trend']

['Intimates' 'Dresses' 'Pants' 'Blouses' 'Knits' 'Outerwear' 'Lounge'
 'Sweaters' 'Skirts' 'Fine gauge' 'Sleep' 'Jackets' 'Swim' 'Trend' 'Jeans'
 'Legwear' 'Shorts' 'Layering' 'Casual bottoms' 'Chemises']
```

```
In [16]: roupas.head()
```

Out[16]:

	Age	Rating	Recommended IND	Division Name	Department Name	Class Name
0	adult	good	yes	Initmates	Intimate	Intimates
1	adult	very good	yes	General	Dresses	Dresses
2	elderly	normal	no	General	Dresses	Dresses
3	adult	very good	yes	General Petite	Bottoms	Pants
4	adult	very good	yes	General	Tops	Blouses

2.4 Algoritmos

2.4.1 Apriori

```
In [11]: encoder = TransactionEncoder()
         roupas_encoded = encoder.fit(roupas_np).transform(roupas_np)
         roupas_encoded = pd.DataFrame(roupas_encoded, columns=encoder.columns_)
```

In [20]:
```
frequent_itemsets = apriori(roupas_encoded, min_support=0.2, use_colnames=True)
association_rules(frequent_itemsets, metric="confidence", min_threshold=0.8).sort_valu
```

Out[20]:

	antecedents	consequents	antecedent support	\
6	(Knits)	(Tops)	0.206331	
34	(adult, Tops, very good)	(yes)	0.207268	
26	(adult, very good)	(yes)	0.486409	
25	(Tops, very good)	(yes)	0.244163	
31	(General, adult, very good)	(yes)	0.280675	
13	(very good)	(yes)	0.558836	
19	(General, very good)	(yes)	0.324301	
12	(good)	(yes)	0.216300	
0	(Dresses)	(adult)	0.269214	
4	(General Petite)	(adult)	0.345944	
21	(yes, General Petite)	(adult)	0.285745	
2	(General)	(adult)	0.590065	
11	(yes)	(adult)	0.822256	
18	(General, yes)	(adult)	0.481979	
27	(yes, very good)	(adult)	0.557771	
9	(very good)	(adult)	0.558836	
28	(very good)	(adult, yes)	0.558836	
32	(General, yes, very good)	(adult)	0.323577	
16	(General, very good)	(adult)	0.324301	
7	(Tops)	(adult)	0.445978	
14	(General, Tops)	(adult)	0.291283	
33	(General, very good)	(adult, yes)	0.324301	
24	(Tops, yes)	(adult)	0.363540	
30	(General, Tops, yes)	(adult)	0.235685	
35	(Tops, yes, very good)	(adult)	0.243737	
22	(Tops, very good)	(adult)	0.244163	
36	(Tops, very good)	(adult, yes)	0.244163	
5	(General Petite)	(yes)	0.345944	
20	(adult, General Petite)	(yes)	0.305854	
10	(adult)	(yes)	0.881646	
3	(General)	(yes)	0.590065	
8	(Tops)	(yes)	0.445978	
17	(adult, General)	(yes)	0.518234	
15	(General, Tops)	(yes)	0.291283	
23	(adult, Tops)	(yes)	0.385651	
1	(Dresses)	(yes)	0.269214	
29	(adult, Tops, General)	(yes)	0.251747	

	consequent support	support	confidence	lift	leverage	conviction
6	0.445978	0.206331	1.000000	2.242262	0.114312	inf
34	0.822256	0.207013	0.998767	1.214666	0.036585	144.120512
26	0.822256	0.485685	0.998511	1.214355	0.085732	119.370574
25	0.822256	0.243737	0.998255	1.214044	0.042972	101.864911
31	0.822256	0.280164	0.998179	1.213951	0.049377	97.581288

13	0.822256	0.557771	0.998094	1.213848	0.098264	93.258562
19	0.822256	0.323577	0.997767	1.213450	0.056918	79.587353
12	0.822256	0.209143	0.966910	1.175922	0.031289	5.371457
0	0.881646	0.242757	0.901725	1.022774	0.005405	1.204312
4	0.881646	0.305854	0.884113	1.002798	0.000853	1.021289
21	0.881646	0.251278	0.879380	0.997429	-0.000648	0.981210
2	0.881646	0.518234	0.878267	0.996167	-0.001994	0.972242
11	0.881646	0.721413	0.877358	0.995136	-0.003526	0.965031
18	0.881646	0.421097	0.873685	0.990970	-0.003837	0.936974
27	0.881646	0.485685	0.870761	0.987653	-0.006072	0.915773
9	0.881646	0.486409	0.870397	0.987241	-0.006286	0.913204
28	0.721413	0.485685	0.869101	1.204721	0.082534	2.128264
32	0.881646	0.280164	0.865833	0.982064	-0.005117	0.882136
16	0.881646	0.280675	0.865476	0.981659	-0.005244	0.879794
7	0.881646	0.385651	0.864731	0.980814	-0.007544	0.874949
14	0.881646	0.251747	0.864268	0.980289	-0.005062	0.871966
33	0.721413	0.280164	0.863899	1.197510	0.046209	2.046917
24	0.881646	0.311861	0.857846	0.973005	-0.008652	0.832574
30	0.881646	0.201985	0.857014	0.972061	-0.005806	0.827728
35	0.881646	0.207013	0.849327	0.963342	-0.007877	0.785501
22	0.881646	0.207268	0.848892	0.962849	-0.007997	0.783240
36	0.721413	0.207013	0.847845	1.175257	0.030870	1.830944
5	0.822256	0.285745	0.825985	1.004535	0.001290	1.021429
20	0.822256	0.251278	0.821563	0.999157	-0.000212	0.996114
10	0.822256	0.721413	0.818256	0.995136	-0.003526	0.977992
3	0.822256	0.481979	0.816823	0.993392	-0.003206	0.970339
8	0.822256	0.363540	0.815151	0.991359	-0.003169	0.961561
17	0.822256	0.421097	0.812562	0.988210	-0.005024	0.948278
15	0.822256	0.235685	0.809127	0.984032	-0.003824	0.931214
23	0.822256	0.311861	0.808661	0.983466	-0.005243	0.928947
1	0.822256	0.217578	0.808197	0.982902	-0.003785	0.926702
29	0.822256	0.201985	0.802335	0.975773	-0.005015	0.899219

2.4.2 Fp-growth

```
In [13]: def sup_item(data, item):
             sup_item = 0
             for transacao in data:
                 if item in transacao:
                     sup_item += 1

             return sup_item
```

```
In [14]: def fp_growth(data, sup=10, conf=10):
             patterns = pyfpgrowth.find_frequent_patterns(data, sup*(len(data)))
             rules = pyfpgrowth.generate_association_rules(patterns, conf)
             list_aux = []
```

```
        for key, value in rules.items():

            try:
                suport_x = patterns[key]/len(data)
            except:
                suport_x = sup_item(data, key)/len(data)

            try:
                suport_y = patterns[value[0]]/len(data)
            except:
                suport_y = sup_item(data, value[0])/len(data)

            conf = value[1]
            suport_xy = conf*suport_x

            try:
                conv = (1-suport_y)/(1-conf)

            except:
                conv = float("inf")

            dict_aux = {'antecedents':key,
                        'consequents':value[0],
                        'antecedent support': suport_x,
                        'consequent support': suport_y,
                        'support': suport_xy,
                        'confidence': conf,
                        'lift': conf/suport_y,
                        'conviction': conv,
                        'leverage': (conf*suport_x)- suport_x*suport_y}
            list_aux.append(dict_aux)
        return pd.DataFrame(list_aux)[['antecedents','consequents','antecedent support','
                                       'support','confidence','lift','leverage','convicti
```

In [15]:
```
fp_growth(roupas_np, 0.2, 0.8)
```

Out[15]:

	antecedents	consequents	antecedent support	\
0	(Knits,)	(Tops,)	0.206331	
1	(good,)	(yes,)	0.216300	
2	(Dresses, General Petite)	(adult,)	0.220603	
3	(General Petite, adult)	(yes,)	0.305854	
4	(General Petite, yes)	(adult,)	0.285745	
5	(Tops, adult, very good)	(yes,)	0.207268	
6	(Tops, very good, yes)	(adult,)	0.243737	
7	(General, Tops, adult)	(yes,)	0.251747	
8	(General, Tops, yes)	(adult,)	0.235685	
9	(Tops, adult)	(yes,)	0.385651	
10	(Tops, yes)	(adult,)	0.363540	

11	(Dresses, adult, very good)	(yes,)	0.258436	
12	(Dresses, very good, yes)	(adult,)	0.289025	
13	(Dresses, General, adult)	(yes,)	0.285191	
14	(Dresses, General, yes)	(adult,)	0.258095	
15	(Dresses, adult)	(yes,)	0.485515	
16	(Dresses, yes)	(adult,)	0.435157	
17	(General, adult, very good)	(yes,)	0.280675	
18	(General, very good, yes)	(adult,)	0.323577	
19	(adult, very good)	(yes,)	0.486409	
20	(very good, yes)	(adult,)	0.557771	
21	(General, adult)	(yes,)	0.518234	
22	(General, yes)	(adult,)	0.481979	
23	(adult,)	(yes,)	0.881646	
24	(yes,)	(adult,)	0.822256	

	consequent support	support	confidence	lift	leverage	conviction
0	0.445978	0.206331	1.000000	2.242262	0.114312	inf
1	0.822256	0.209143	0.966910	1.175922	0.031289	5.371457
2	0.881646	0.200324	0.908073	1.029974	0.005830	1.287470
3	0.822256	0.251278	0.821563	0.999157	-0.000212	0.996114
4	0.881646	0.251278	0.879380	0.997429	-0.000648	0.981210
5	0.822256	0.207013	0.998767	1.214666	0.036585	144.120512
6	0.881646	0.207013	0.849327	0.963342	-0.007877	0.785501
7	0.822256	0.201985	0.802335	0.975773	-0.005015	0.899219
8	0.881646	0.201985	0.857014	0.972061	-0.005806	0.827728
9	0.822256	0.311861	0.808661	0.983466	-0.005243	0.928947
10	0.881646	0.311861	0.857846	0.973005	-0.008652	0.832574
11	0.822256	0.258010	0.998351	1.214161	0.045509	107.819325
12	0.881646	0.258010	0.892689	1.012525	0.003192	1.102901
13	0.822256	0.230999	0.809979	0.985069	-0.003501	0.935390
14	0.881646	0.230999	0.895015	1.015163	0.003450	1.127338
15	0.822256	0.391275	0.805897	0.980104	-0.007943	0.915718
16	0.881646	0.391275	0.899158	1.019863	0.007620	1.173656
17	0.822256	0.280164	0.998179	1.213951	0.049377	97.581288
18	0.881646	0.280164	0.865833	0.982064	-0.005117	0.882136
19	0.822256	0.485685	0.998511	1.214355	0.085732	119.370574
20	0.881646	0.485685	0.870761	0.987653	-0.006072	0.915773
21	0.822256	0.421097	0.812562	0.988210	-0.005024	0.948278
22	0.881646	0.421097	0.873685	0.990970	-0.003837	0.936974
23	0.822256	0.721413	0.818256	0.995136	-0.003526	0.977992
24	0.881646	0.721413	0.877358	0.995136	-0.003526	0.965031

Regras2

June 5, 2019

1 0. Introdução

Trabalho:
Aluno: Maicon Dall'Agnol
R.A.: 151161868
Disciplina: Tópico em Aprendizado de Máquina
Objetivos :

- Escolha dois datasets específicos para a tarefa de Regras de Associação. Não tente "produzir" um dataset como esse. Possivelmente, você irá falhar nisso. O dataset para Regras de Associação deve conter transações.
- Para cada dataset, você irá aplicar os algoritmos APriori e FP-Growth. Seu objetivo é minerar regras que contenham informações relevantes no dataset, seja porque alguma combinação de itens aparece com muita frequência, seja porque alguma combinação de itens não aparece com muita frequência mas está se destacando. Para isso, você deve variar os parâmetros de suporte, confiança e lift.
- Existem duas métricas não estudadas em sala de aula, 'leverage' e 'conviction', que estão disponibilizadas no pacote mlxtend para regras de associação. Estude essas métricas e explique como elas podem contribuir para as análises dos seus datasets. (Veja: http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/)
- No relatório, você deverá explicar como usou os parâmetros para a mineração e o que foi obtido. As regras selecionadas devem ser exibidas com suas medidas equivalentes (suporte, confiança, lift, leverage, conviction). Mostre as regras que você considerou mais relevantes e justifique por quê.
- Compare os resultados gerados pelos dois algoritmos. Conclua sobre as diferenças encontradas nos resultados de cada dataset na aplicação dos dois algoritmos.

1.1 0.1 Dependências

Para realização da tarefa foram utilizados as seguintes bibliotecas:

```
In [5]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
```

```
import seaborn as sns
import pandas_profiling

# Encoder
from mlxtend.preprocessing import TransactionEncoder

# Algoritmos
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
import pyfpgrowth

#Metricas
from mlxtend.frequent_patterns import association_rules

import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

2 1. Dados

Este é um conjunto de dados educacionais que é coletado do sistema de gerenciamento de aprendizado (LMS) chamado Kalboard 360

2.1 1.1 Informações sobre os dados:

Atributos:

- gender - sexo do aluno (nominal: 'Masculino' ou 'Feminino')
- NationalITy - nacionalidade do estudante (nominal: 'Kuwait', 'Líbano', 'Egipto', 'Arábia Saudita', 'EUA', 'Jordânia', 'Venezuela', 'Irão', 'Tunes', 'Marrocos', 'Síria',' Palestina ',' Iraque ',' Líbia ')
- PlaceofBirth - Local de nascimento do estudante (nominal: 'Kuwait', 'Líbano', 'Egipto', 'Arábia Saudita', 'EUA', 'Jordânia', 'Venezuela', 'Irão', 'Tunes', 'Marrocos',' Síria ',' Palestina ',' Iraque ',' Líbia ')
- StageID - estudante de nível educacional pertence (nominal: 'nível inferior', 'MiddleSchool', 'HighSchool')
- GradeID - estudante de graduação pertence (nominal: 'G-01', 'G-02', 'G-03', 'G-04', 'G-05', 'G-06', 'G-07' ',' G-08 ',' G-09 ',' G-10 ',' G-11 ',' G-12 ')
- SectionID - aluno de sala de aula pertence (nominal: 'A', 'B', 'C')
- Topic - tópico do curso (nominal: inglês, espanhol, francês, árabe, IT, matemática, química, biologia, ciência, história, Alcorão ',' Geologia ')
- Semester - ano letivo (nominal: 'Primeiro', 'Segundo')
- Relation - responsável pelo aluno (nominal: 'mãe', 'pai')

- raisedhands - quantas vezes o aluno levanta a mão na sala de aula (numérico: 0-100)
- VisITedResources - quantas vezes o aluno visita o conteúdo do curso (numérico: 0-100)
- AnnouncementsView - quantas vezes o aluno verifica os novos anúncios (numérico: 0-100)
- Discussion - quantas vezes o aluno participa de grupos de discussão (numérico: 0-100)
- ParentAnsweringSurvey - pai respondeu as pesquisas que são fornecidas a partir da escola ou não (nominal: 'Sim', 'Não')
- ParentschoolSatisfaction - o Grau de satisfação dos pais da escola (nominal: 'Sim', 'Não')
- StudentAbsenceDays - o número de dias de ausência para cada aluno (nominal: acima de 7, sub-7)

Classe

- Class

2.2 Importando Dataset

```
In [27]: edu = pd.read_csv('xAPI-Edu-Data.csv')

In [28]: edu.head()
```

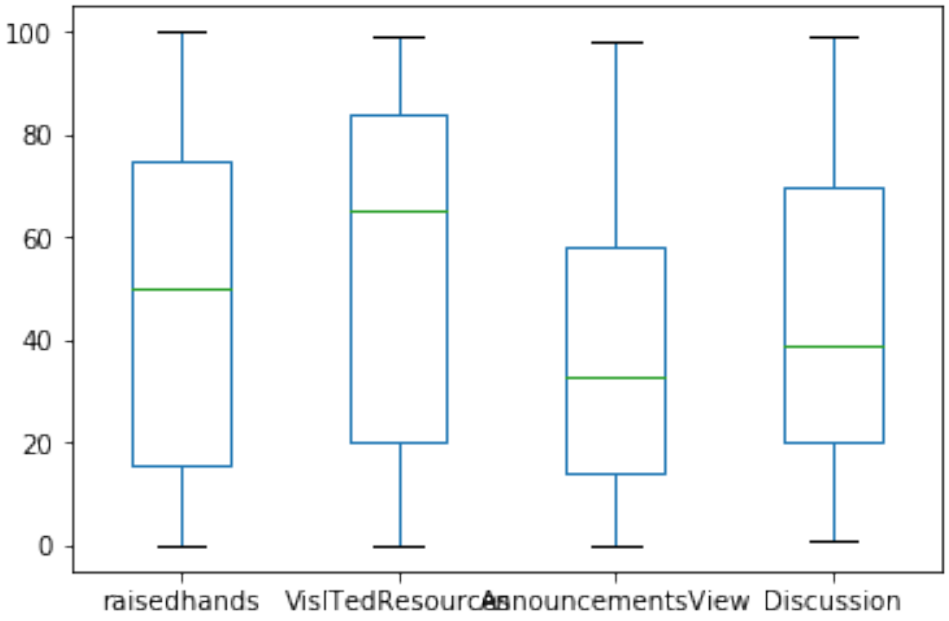
Out[28]:

	gender	NationalITy	PlaceofBirth	StageID	GradeID	SectionID	Topic	Semester	Relation	raisedhands	VisITedResources	AnnouncementsView	Discussion	ParentAnsweringSurvey	ParentschoolSatisfaction	StudentAbsenceDays	Class
0	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	15	16	2	20	Yes	Good	Under-7	M
1	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	20	20	3	25	Yes	Good	Under-7	M
2	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	10	7	0	30	No	Bad	Under-7	M
3	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	30	25	5	35	No	Bad	Under-7	M
4	M	KW	KuwaIT	lowerlevel	G-04	A	IT	F	Father	40	50	12	50	No	Bad	Under-7	M

```
2            Above-7     L
3            Above-7     L
4            Above-7     M
```

In [29]:
```
edu.plot.box()
```

Out[29]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f258f84ca90>
```



In [30]:
```
edu.isna().sum()
```

Out[30]:
```
gender                      0
NationalITy                 0
PlaceofBirth                0
StageID                     0
GradeID                     0
SectionID                   0
Topic                       0
Semester                    0
Relation                    0
raisedhands                 0
VisITedResources            0
AnnouncementsView           0
Discussion                  0
ParentAnsweringSurvey       0
ParentschoolSatisfaction    0
StudentAbsenceDays          0
Class                       0
dtype: int64
```

2.3 Discretizando

2.4 Algoritmos

```
In [31]: def list_attrib_num(data):
             num = []

             for atributo, tipo in zip(data.columns, data.dtypes):
                 if tipo != object:
                     num.append(atributo)

             return num
```

```
In [32]: for coluna in list_attrib_num(edu):
             under = 'under_mean_' + coluna
             over = 'over_mean_' + coluna
             edu[coluna] = [under if x < edu[coluna].mean() else over for x in edu[coluna]]
```

```
In [33]: edu.gender = ['male' if x == 'M' else 'female' for x in edu.gender]
```

```
In [34]: edu.head()
```

Out[34]:

	gender	NationalITy	PlaceofBirth	StageID	GradeID	SectionID	Topic	\
0	male	KW	KuwaIT	lowerlevel	G-04	A	IT	
1	male	KW	KuwaIT	lowerlevel	G-04	A	IT	
2	male	KW	KuwaIT	lowerlevel	G-04	A	IT	
3	male	KW	KuwaIT	lowerlevel	G-04	A	IT	
4	male	KW	KuwaIT	lowerlevel	G-04	A	IT	

	Semester	Relation	raisedhands	VisITedResources	\
0	F	Father	under_mean_raisedhands	under_mean_VisITedResources	
1	F	Father	under_mean_raisedhands	under_mean_VisITedResources	
2	F	Father	under_mean_raisedhands	under_mean_VisITedResources	
3	F	Father	under_mean_raisedhands	under_mean_VisITedResources	
4	F	Father	under_mean_raisedhands	under_mean_VisITedResources	

	AnnouncementsView	Discussion	ParentAnsweringSurvey	\
0	under_mean_AnnouncementsView	under_mean_Discussion	Yes	
1	under_mean_AnnouncementsView	under_mean_Discussion	Yes	
2	under_mean_AnnouncementsView	under_mean_Discussion	No	
3	under_mean_AnnouncementsView	under_mean_Discussion	No	
4	under_mean_AnnouncementsView	over_mean_Discussion	No	

	ParentschoolSatisfaction	StudentAbsenceDays	Class
0	Good	Under-7	M
1	Good	Under-7	M
2	Bad	Above-7	L
3	Bad	Above-7	L
4	Bad	Above-7	M

```
In [35]: edu_np = edu.to_numpy()
```

2.4.1 Apriori

```
In [36]: encoder = TransactionEncoder()
         edu_encoded = encoder.fit(edu_np).transform(edu_np)
         edu_encoded = pd.DataFrame(edu_encoded, columns=encoder.columns_)

In [52]: frequent_itemsets = apriori(edu_encoded, min_support=0.35, use_colnames=True)
         association_rules(frequent_itemsets, metric="confidence", min_threshold=0.8).sort_valu

CPU times: user 31.1 ms, sys: 0 ns, total: 31.1 ms
Wall time: 31.1 ms
```

2.4.2 Fp-growth

```
In [44]: def sup_item(data, item):
             sup_item = 0
             for transacao in data:
                 if item in transacao:
                     sup_item += 1

             return sup_item

In [45]: def fp_growth(data, sup=0.1, conf=0.1):
             patterns = pyfpgrowth.find_frequent_patterns(data, sup*(len(data)))
             rules = pyfpgrowth.generate_association_rules(patterns, conf)
             list_aux = []

             for key, value in rules.items():

                  if value[0] is not ():
                     try:
                         suport_x = patterns[key]/len(data)
                     except:
                         suport_x = sup_item(data, key)/len(data)

                     try:
                         suport_y = patterns[value[0]]/len(data)

                     except:
                         suport_y = sup_item(data, value[0])/len(data)

                     conf = value[1]
                     suport_xy = conf*suport_x

                     try:
                         conv = (1-suport_y)/(1-conf)

                     except:
```

```
                    conv = float("inf")


                dict_aux = {'antecedents':key,
                            'consequents':value[0],
                            'antecedent support': suport_x,
                            'consequent support': suport_y,
                            'support': suport_xy,
                            'confidence': conf,
                            'lift': conf/suport_y,
                            'conviction': conv,
                            'leverage': (conf*suport_x)- suport_x*suport_y}
                list_aux.append(dict_aux)
            return pd.DataFrame(list_aux)[['antecedents','consequents','antecedent support','c
                                           'support','confidence','lift','leverage','convictio
```

In [53]:
```
fp_growth(edu_np, 0.35, 0.8)
```

Out[53]:

	antecedents	consequents	antecedent support	consequent support
0	(KW,)	(KuwaIT,)	0.372917	0.375000
1	(KuwaIT,)	(KW,)	0.375000	0.372917
2	(Mum,)	(Jordan,)	0.410417	0.725000
3	(M,)	(Jordan,)	0.439583	0.725000
4	(Jordan, over_mean_raisedhands)	(over_mean_VisITedResources,)	0.387500	0.564583
5	(over_mean_VisITedResources, over_mean_raisedh...	(Under-7,)	0.443750	0.602083
6	(Under-7, over_mean_VisITedResources)	(Jordan,)	0.447917	0.725000
7	(Under-7, over_mean_raisedhands)	(over_mean_VisITedResources,)	0.410417	0.564583
8	(A, MiddleSchool)	(Jordan,)	0.377083	0.725000
9	(Jordan, MiddleSchool)	(A,)	0.402083	0.589583
10	(Jordan, Yes)	(over_mean_VisITedResources,)	0.397917	0.564583
11	(Yes, over_mean_VisITedResources)			
12	(Good, over_mean_VisITedResources)			
13	(Good, Jordan)			
14	(Jordan, over_mean_VisITedResources)			
15	(Jordan, Under-7)			

11	(Good,)	0.408333	0.608333
12	(Jordan,)	0.431250	0.725000
13	(over_mean_VisITedResources,)	0.470833	0.564583
14	(Good,)	0.502083	0.608333
15	(over_mean_VisITedResources,)	0.425000	0.564583

	support	confidence	lift	leverage	conviction
0	0.368750	0.988827	2.636872	0.228906	55.937500
1	0.368750	0.983333	2.636872	0.228906	37.625000
2	0.387500	0.944162	1.302293	0.089948	4.925000
3	0.356250	0.810427	1.117830	0.037552	1.450625
4	0.366667	0.946237	1.675991	0.147891	8.098750
5	0.366667	0.826291	1.372387	0.099492	2.290709
6	0.375000	0.837209	1.154771	0.050260	1.689286
7	0.366667	0.893401	1.582408	0.134952	4.084623
8	0.352083	0.933702	1.287864	0.078698	4.147917
9	0.352083	0.875648	1.485197	0.115022	3.300434
10	0.350000	0.879581	1.557930	0.125343	3.615851
11	0.362500	0.887755	1.459323	0.114097	3.489394
12	0.414583	0.961353	1.326004	0.101927	7.115625
13	0.414583	0.880531	1.559612	0.148759	3.644599
14	0.414583	0.825726	1.357358	0.109149	2.247421
15	0.375000	0.882353	1.562839	0.135052	3.701042

In [ ]: